# Euler and improved Euler using SAGE

The goal is to find an approximate solution to the problem

$$y' = f(x, y), \quad y(a) = c, \tag{1}$$

where $f(x, y)$ is some given function. We shall try to approximate the value of the solution at $x = b$, where $b > a$ is given. This will be explained mathematically and using the software package SAGE, available from `http://sage.scipy.org`. To use SAGE below on you machine, you will have to (a) install SAGE, (b) "attach" the file `eulers_method.sage` file in the examples subdirectory.

The basic idea can also be explained "algebraically". Recall from the definition of the derivative in calculus 1 that

$$y'(x) \cong \frac{y(x+h) - y(x)}{h},$$

$h > 0$ is a given and small. This an the DE together give $f(x, y(x)) \cong \frac{y(x+h)-y(x)}{h}$. Now solve for $y(x + h)$:

$$y(x + h) \cong y(x) + h \cdot f(x, y(x)).$$

If we call $h \cdot f(x, y(x))$ the "correction term" (for lack of anything better), call $y(x)$ the "old value of $y$", and call $y(x + h)$ the "new value of $y$", then this approximation can be re-expressed

$$y_{new} = y_{old} + h \cdot f(x, y_{old}).$$

**Tabular idea**: Let $n > 0$ be an integer, which we call the **step size**. This is related to the increment by

$$h = \frac{b - a}{n}.$$

This can be expressed simplest using a table.

| $x$ | $y$ | $hf(x, y)$ |
|---|---|---|
| $a$ | $c$ | $hf(a, c)$ |
| $a + h$ | $c + hf(a, c)$ | $\vdots$ |
| $a + 2h$ | $\vdots$ | |
| $\vdots$ | | |
| $b$ | ??? | xxx |

The goal is to fill out all the blanks of the table but the xxx entry and find the ??? entry, which is the **Euler's method approximation for** $y(b)$.

Improved Euler's method

**Geometric idea**: The basic idea can be easily expressed in geometric terms. As in Euler's method, we know the solution must go through the point $(a, c)$ and we know its slope there is $m = f(a, c)$. If we went out one step using the tangent line approximation to the solution curve, the approximate slope to the tangent line at $x = a + h, y = c + h \cdot f(a, c)$ would be $m' = f(a + h, c + h \cdot f(a, c))$. The idea is that instead of using $m = f(a, c)$ as the slope of the line to get our first approximation, use $\frac{m+m'}{2}$. The "improved" tangent-line approximation at $(a, c)$ is:

$$y(a + h) \cong c + h \cdot \frac{m + m'}{2} = c + h \cdot \frac{f(a, c) + f(a + h, c + h \cdot f(a, c))}{2}.$$

(This turns out to be a better apprpximation than the tangent-line approximation $y(a + h) \cong c + h \cdot f(a, c)$ used in Euler's method.) Now we know the solution passes through a point which is "nearly" equal to $(a + h, c + h \cdot \frac{m+m'}{2})$. We now repeat this tangent-line approximation with $(a, c)$ replaced by $(a + h, c + h \cdot f(a, c)$. Keep repeating this number-crunching at $x = a$, $x = a + h$, $x = a + 2h$, ..., until you get to $x = b$.

**Tabular idea**: The integer step size $n > 0$ is related to the increment by

$$h = \frac{b - a}{n},$$

as before.

The improved Euler method can be expressed simplest using a table.

| $x$ | $y$ | $h\frac{m+m'}{2} = h\frac{f(x,y)+f(x+h,y+h\cdot f(x,y))}{2}$ |
|---|---|---|
| $a$ | $c$ | $h\frac{f(a,c)+f(a+h,c+h\cdot f(a,c))}{2}$ |
| $a + h$ | $c + h\frac{f(a,c)+f(a+h,c+h\cdot f(a,c))}{2}$ | $\vdots$ |
| $a + 2h$ | $\vdots$ | |
| $\vdots$ | | |
| $b$ | ??? | xxx |

The goal is to fill out all the blanks of the table but the xxx entry and find the ??? entry, which is the **improved Euler's method approximation for** $y(b)$.

The idea for systems of ODEs is similar. This is implemented below as well.

- The `SAGE` program `improved_eulers_method` implements Improved Euler's method for finding numerically the solution of the 1st order ODE $y' = f(x, y)$, $y(a) = c$. The "x" column of the table increments from x0 to x1 by h (so (x1-x0)/h must be an integer). In the "y" column, the new y-value equals the old y-value plus the corresponding entry in the last column.

  Here is how the syntax of the program works:

```
sage: RR = RealField(sci_not=0, prec=4, rnd='RNDU')
sage: x,y = PolynomialRing(RR,2).gens()
sage: improved_eulers_method(5*x+y-5,0,1,1/2,1)
 x                          y                  (h/2)*(f(x,y)+f(x+h,y+h*f(x,y))
 0                          1                       -1.87
1/2                    -0.875                      -1.37
 1                      -2.25                      -0.687
sage: x,y=PolynomialRing(QQ,2).gens()
sage: improved_eulers_method(5*x+y-5,0,1,1/2,1)
 x                          y                  (h/2)*(f(x,y)+f(x+h,y+h*f(x,y))
 0                          1                       -15/8
1/2                     -7/8                       -95/64
 1                    -151/64                    -435/512
```

- The `SAGE` program `eulers_method` implements Euler's method for finding numerically the solution of the 1st order ODE $y' = f(x, y)$, $y(a) = c$. The "x" column of the table increments from x0 to x1 by h (so (x1-x0)/h must be an integer). In the "y" column, the new y-value equals the old y-value plus the corresponding entry in the last column.

  Here is how the syntax of the program works:

```
sage: x,y=PolynomialRing(QQ,2).gens()
sage: eulers_method(5*x+y-5,0,1,1/2,1)
 x                  y                h*f(x,y)
 0                  1                  -2
1/2                -1                -7/4
 1              -11/4               -11/8
sage: RR = RealField(sci_not=0, prec=4, rnd='RNDU')
sage: x,y=PolynomialRing(RR,2).gens()
sage: eulers_method(5*x+y-5,0,1,1/2,1)
 x                  y                h*f(x,y)
 0                  1               -2.00
1/2              -1.00               -1.75
 1               -2.75               -1.37
```

- The `SAGE` program `eulers_method_2x2` implements Euler's method for finding numerically the solution of the 1st order system of two ODEs

$$x' = f(t,x,y), x(t0) = x0. y' = g(t,x,y), y(t0) = y0.$$

The "t" column of the table increments from t0 to t1 by h (so $(t1-t0)/h$ must be an integer). In the "x" column, the new x-value equals the old x-value plus the corresponding entry in the next (third) column. In the "y" column, the new y-value equals the old y-value plus the corresponding entry in the next (last) column.

Here is how the syntax of the program works:

To approximate $y(1)$ using 3 steps, where

$$\begin{aligned} x' &= x + y + t, \qquad & x(0) = 0, \\ y' &= x - y, & y(0) = 0, \end{aligned}$$

use the following `SAGE` commands:

```
sage: t, x, y = PolynomialRing(QQ,3).gens()
sage: f = x+y+t; g = x-y
sage: eulers_method_2x2(f,g, 0, 0, 0, 1/3, 1)
 t                   x               h*f(t,x,y)                 y          h*g(t,x,y)
 0                   0                        0                 0                   0
1/3                  0                      1/9                 0                   0
2/3                1/9                      7/27                0                1/27
 1                10/27                    38/81              1/27                1/9
sage: RR = RealField(sci_not=0, prec=4, rnd='RNDU')
sage: t,x,y=PolynomialRing(RR,3).gens()
sage: f = x+y+t; g = x-y
sage: eulers_method_2x2(f,g, 0, 0, 0, 1/3, 1)
 t                   x               h*f(t,x,y)                 y          h*g(t,x,y)
 0                   0                    0.000                 0               0.000
1/3               0.000                    0.125             0.000               0.000
2/3               0.125                    0.282             0.000              0.0430
 1                0.407                    0.563            0.0430               0.141
```

To numerically approximate y(1), where

$$(1+t^2)y'' + y' - y = 0, y(0) = 1, y'(0) = -1,$$

using 4 steps of Euler's method, first convert to a system: $y1' = y2, y1(0) = 1$; $y2' = (y1 - y2)/(1+t^2), y2(0) = -1$.

```
sage: RR = RealField(sci_not=0, prec=4, rnd='RNDU')
sage: t, y1, y2=PolynomialRing(RR,3).gens()
sage: f = y2; g = (y1-y2)/(1+t^2)
sage: eulers_method_2x2(f,g, 0, 1, -1, 1/4, 1)
 t                   x               h*f(t,x,y)                 y          h*g(t,x,y)
 0                   1                   -0.250                -1               0.500
1/4               0.750                   -0.125            -0.500               0.282
1/2               0.625                  -0.0546            -0.218               0.188
3/4               0.625                 -0.00781           -0.0312               0.110
 1                0.625                    0.0196            0.0782              0.0704
```

To numerically approximate y(1), where $y'' + ty' + y = 0$, $y(0) = 1$, $y'(0) = 0$:

```
sage: t,x,y=PolynomialRing(RR,3).gens()
sage: f = y2; g = -y1-y2*t
sage: eulers_method_2x2(f,g, 0, 1, 0, 1/4, 1)
 t                  x              h*f(t,x,y)                  y          h*g(t,x,y)
 0                  1                   0.000                  0              -0.250
1/4               1.00                 -0.0625             -0.250              -0.234
1/2              0.938                  -0.117             -0.468              -0.171
3/4              0.875                  -0.156             -0.625              -0.101
 1               0.750                  -0.171             -0.687             -0.0156
```